

Curso Técnico Superior Profissional em: Tecnologias e Programação de Sistemas de Informação

2.º Ano/ 1.º Semestre

Unidade Curricular: Arquitetura de Dispositivos

Docente: David Jardim

FICHA DE TRABALHO 4

Exercícios:

1. Crie um projeto com o nome Aula4.
2. Crie uma classe denominada por **MatrixUtilities** e implemente as seguintes funções:
 - a. **show** – imprime na consola uma matriz tendo em conta as linhas e as colunas da mesma.
 - b. **isMatrix** – verifica se o array multi-dimensional que é passado como argumento é uma matriz, ou seja, o número de colunas deve ser igual para todas as linhas.
 - c. **isIdentity** – verifica se o array multi-dimensional que é passado como argumento é uma matriz identidade, ou seja, os elementos na diagonal da matriz são iguais a 1 e os restantes iguais a zero.

$$I_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}_{n \times n}$$

Figura 1 - Matriz Identidade

- d. **multiplyBy** – pretende-se multiplicar uma matriz por um valor constante e devolver o resultado. A matriz e a constante são passadas como argumentos.
- e. **areCompatibleForSum** – verifica se dois arrays multi-dimensionais (passados como argumentos) são compatíveis para efetuar uma soma. Duas matrizes apenas podem ser adicionadas se tiverem o mesmo tamanho.
- f. **sumOf** – dado dois arrays multi-dimensionais que são passados como argumentos, efetue a soma entre eles e devolva o resultado verificando previamente se é possível efetuar a soma.

$$\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} + \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} = \begin{bmatrix} a_1 + a_2 & b_1 + b_2 \\ c_1 + c_2 & d_1 + d_2 \end{bmatrix}$$

Figura 2 - Operação de adição entre matrizes

Cofinanciado por:

3. Crie uma classe denominada por **CharacterDrawingUtilities** e implemente as seguintes funções:
- a. **drawElement** – imprime um determinado caractere passado como argumento na mesma linha.
 - b. **drawNewLine** – imprime uma nova linha vazia.
 - c. **drawHorizontalSegmentWith** – imprime uma linha horizontal com um determinado caractere passado como argumento (utilize a função 3.a).
 - d. **drawFilledRectangleWith** – imprime um retângulo preenchido onde a altura, largura e o caractere a desenhar são passados como argumentos (utilize as funções 3.b e 3C).

```
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
+ + + + + + + + +
```

- e. **drawEmptyRectangleWith** – imprime um retângulo vazio onde a altura, largura e o caractere a desenhar são passados como argumentos (utilize as funções 3.a e 3b).

```
* * * * *
*           *
*           *
*           *
*           *
*           *
*           *
*           *
*           *
*           *
* * * * *
```

Cofinanciado por: